1 Enviando solicitudes con XMLHttpRequest

Desde el momento que tenemos creado nuestro objeto XMLHttpRequest podemos comenzar el ciclo de peticiones/respuestas.

Recordad que **el único propósito de este objeto es enviar solicitudes y recibir respuestas**.

Cualquier otra cosa como ? cambiar el interfaz de usuario, intercambiar imágenes, interpretar los datos recibidos, etc.. ? es trabajo de **Javascript, CSS** o bien cualquier otro código que tengamos en nuestras páginas.

1.1 Sumario

1.2 Bienvenido a la Caja Negra.

Ajax dispone de un **modelo de seguridad**. Como resultado de dicho modelo (y específicamente del objeto XMLHttpRequest) **solamente se podrán realizar solicitudes al mismo dominio dónde se está ejecutando la aplicación Ajax**.

1.3 Indicando la URL del servidor

Lo primero que debemos conocer es la URL del servidor al cuál nos queremos conectar y dónde se encuentran nuestras páginas php.

El listado 6 muestra cómo construir una URL en la que se envían datos formando parte de dicha URL (método GET).

1.4 Listado 6. Construyendo una URL

```
<script language="javascript" type="text/javascript">
   var request = false;
   try {
     request = new XMLHttpRequest();
   } catch (trymicrosoft) {
     try {
       request = new ActiveXObject("Msxml2.XMLHTTP");
     } catch (othermicrosoft) {
       try {
         request = new ActiveXObject("Microsoft.XMLHTTP");
       } catch (failed) {
         request = false;
       }
     }
   }

   if (!request)
     alert("Error inicializando XMLHttpRequest!");

   function obtenerInfoCliente() {
     var phone = document.getElementById("telefono").value;
     var url = "buscarcliente.php?telefono=" + escape(telefono);
   }
</script>
```

1.5 Sentencia escape ()

Cuando se envían datos por el método GET el script recibe los datos como una cadena adjunta a la URL original, en la cual algunas secuencias de caracteres han de interpretarse como caracteres especiales, como el espacio en blanco o el separador entre variable y valor.

Los caracteres escritos en dicha notación se llaman sequenze escape y utilizan la codificación URL en la cual:

1. los pares nombre=valor están separados por &;
2. los espacios se sustituyen con +;
3. los caracteres alfanuméricos son sustituidos por el equivalente hexadecimal precedido de %.

Empleando la sentencia escape(valor) convertiremos al formato de la URL los caracteres especiales que estén dentro de dicho valor.

Veamos el listado:

El código anterior creará una variable llamada phone que contendrá el valor del campo teléfono de nuestro formulario.

Véase el listado 7 del formulario. Fijarse en los id de los campos..

1.6 Listado 7. El formulario

```
<body>
  <p><img src="pizzahut.jpg" alt="Pizza Hut Pizza" /></p>
  <form action="POST">
   <p>Introduzca su n&uacute;mero:
     <input type="text" size="14" name="telefono" id="telefono"
           onChange="obtenerInfoCliente();" />
   </p>
   <p>Su pizza ser&aacute; entregada en la direcci&oacute;n:</p>
   <div id="address"></div>
   <p>Escriba su solicitud aqu&iacute;:</p>
   <p><textarea name="solicitud" rows="6" cols="50" id="solicitud"></textarea>
   </p>
   <p><input type="submit" value="Pedir una Pizza" id="submit" />
   </p>
  </form>
 </body>
```

Cuando los clientes introducen su teléfono o modifican el número del mismo se llamará a la función **obtenerInfoCliente()** mostrada en el listado 7.

Esta function lo que realiza es capturar el teléfono que tenemos en ese campo del formulario y lo utiliza para construir la URL que llamará a nuestra aplicación php que se encarga de buscar la dirección del cliente.

Recordar una vez más que Ajax solamente podrá conectar con URL almacenadas en el mismo dominio por lo que no es necesario indicar el dominio en la URL.

Y recordamos una vez más que el método escape() se emplea para codificar caracteres especiales como espacios en blanco (que los convertirá en (%20), para que puedan ser enviados correctamente en la URL.

Podríamos añadir todos los parámetros que queramos a la URL simplemente separando por &parámetro2=valor&parámetro3=valor? Indicar que el primer parámetro irá separa por ? En vez de &.

1.7 Ejecutando la solicitud

Una vez que ya tenemos la URL codificada, podemos configurar su ejecución mediante el **método open(**) del objeto **XMLHttpRequest**.

Este método tiene hasta 5 parámetros posibles:

- **request-type**: El tipo de solicitud a enviar. Valores típicos son GET o POST, aunque se pueden enviar cabeceras HEAD.
- **url**: La URL a la que queremos conectar..
- **asynch**: true si queremos que sea una solicitud asíncrona y false si queremos que sea síncrono. Por defecto es true.
- **username**: Si se necesita autenticación para acceder a la página php. Es un parámetro opcional.
- **password**: Contraseña para la autenticación anterior.

Tipicalmente veremos solamente los 3 primeros parámetros. De hecho usaremos casi siempre los tres primeros.

Veamos el código con el método open en el listado 8.

1.8 Listado 8. Abrimos la solicitud

```
function obtenerInfoCliente() {
     var phone = document.getElementById("phone").value;
     var url = " buscarcliente.php?telefono=" + escape(phone);
     request.open("GET", url, true);
   }
```